

# (Generalized) Post Correspondence Problem and semi-Thue systems

François Nicolas\*

November 12, 2008

## Abstract

Let  $\text{PCP}(k)$  denote the following restriction of the well-known Post Correspondence Problem [10]: given alphabet  $\Sigma$  of cardinality  $k$  and two morphisms  $\sigma, \tau : \Sigma^* \rightarrow \{0, 1\}^*$ , decide whether there exists  $w \in \Sigma^+$  such that  $\sigma(w) = \tau(w)$ . Let  $\text{ACCESSIBILITY}(k)$  denote the following restriction of the accessibility problem for semi-Thue systems: given a  $k$ -rule semi-Thue system  $T$  and two words  $u$  and  $v$ , decide whether  $v$  is derivable from  $u$  modulo  $T$ . In 1980, Claus showed that if  $\text{ACCESSIBILITY}(k)$  is undecidable then  $\text{PCP}(k + 4)$  is also undecidable [2]. The aim of the paper is to present a clean, detailed proof of the statement.

We proceed in two steps, using the Generalized Post Correspondence Problem [4] as an auxiliary. Let  $\text{GPCP}(k)$  denote the following restriction of GPCP: given an alphabet  $\Sigma$  of cardinality  $k$ , two morphisms  $\sigma, \tau : \Sigma^* \rightarrow \{0, 1\}^*$  and four words  $s, t, s', t' \in \{0, 1\}^*$ , decide whether there exists  $w \in \Sigma^*$  such that  $s\sigma(w)t = s'\tau(w)t'$ . First, we prove that if  $\text{ACCESSIBILITY}(k)$  is undecidable then  $\text{GPCP}(k + 2)$  is also undecidable. Then, we prove that if  $\text{GPCP}(k)$  is undecidable then  $\text{PCP}(k + 2)$  is also undecidable. (The latter result can also be found in [7].)

To date, the sharpest undecidability bounds for both PCP and GPCP have been deduced from Claus's result: since Matiyasevich and Sénizergues showed that  $\text{ACCESSIBILITY}(3)$  is undecidable [9],  $\text{GPCP}(5)$  and  $\text{PCP}(7)$  are undecidable.

## 1 Introduction

A *word* is a finite sequence of *letters*. The *empty word* is denoted by  $\varepsilon$ . For every word  $w$ , the *length* of  $w$  is denoted by  $|w|$ . A set of words is called a *language*. Word concatenation is denoted multiplicatively. For every language  $L$ ,  $L^+$  denotes the closure of  $L$  under concatenation, and  $L^*$  denotes the language  $L^+ \cup \{\varepsilon\}$ . An *alphabet* is a finite set of letters. For every alphabet  $\Sigma$ ,  $\Sigma^+$  equals the set of all non-empty words over  $\Sigma$ , and  $\Sigma^*$  equals the set of all words over  $\Sigma$  including the empty word.

---

\*E-mail address: nicolas@cs.helsinki.fi

Let  $x$  and  $y$  be two words. We say that  $x$  is a *prefix* (resp. *suffix*) of  $y$  if there exists a word  $z$  such that  $xz = y$  (resp.  $zx = y$ ). A prefix (resp. suffix) of  $y$  is called *proper* if it is distinct from  $y$ . We say that  $x$  *occurs* in  $y$  if there exists a word  $z$  such that  $zx$  is a prefix of  $y$ . The number of occurrences of  $x$  in  $y$  is denoted by  $|y|_x$ :  $|y|_x$  equals the number of words  $z$  such that  $zx$  is a prefix of  $y$ .

## 1.1 The (Generalized) Post Correspondence Problem

Let  $\Sigma$  and  $\Delta$  be alphabets. A function  $\sigma : \Sigma^* \rightarrow \Delta^*$  is called a *morphism* if  $\sigma(xy) = \sigma(x)\sigma(y)$  for every  $x, y \in \Sigma^*$ . Note that any morphism maps the empty word to itself. Moreover, for every function  $\sigma_1 : \Sigma \rightarrow \Delta^*$ , there exists exactly one morphism  $\sigma : \Sigma^* \rightarrow \Delta^*$  such that  $\sigma(a) = \sigma_1(a)$  for every  $a \in \Sigma$ . Hence, although the set of all functions from  $\Sigma^*$  to  $\Delta^*$  has the power of the continuum, the restriction of  $\sigma$  to  $\Sigma$  provides a finite encoding of  $\sigma$  for every morphism  $\sigma : \Sigma^* \rightarrow \Delta^*$ . From now on such encodings are considered as canonical.

The well-known Post Correspondence Problem (PCP) [10] can be stated as follows: given an alphabet  $\Sigma$  and two morphisms  $\sigma, \tau : \Sigma^* \rightarrow \{0, 1\}^*$ , decide whether there exists  $w \in \Sigma^+$  such that  $\sigma(w) = \tau(w)$ . For each integer  $k \geq 1$ , PCP( $k$ ) denotes the restriction of PCP to instances  $(\Sigma, \sigma, \tau)$  such that  $\Sigma$  has cardinality  $k$ .

The Generalized Post Correspondence Problem (GPCP) [4] is: given an alphabet  $\Sigma$ , two morphisms  $\sigma, \tau : \Sigma^* \rightarrow \{0, 1\}^*$  and four words  $s, t, s', t' \in \{0, 1\}^*$ , decide whether there exists  $w \in \Sigma^*$  such that  $s\sigma(w)t = s'\tau(w)t'$ . Note that if  $st = s't'$  then  $\varepsilon$  is a feasible solution of GPCP on  $(\Sigma, \sigma, \tau, s, t, s', t')$ , while all feasible solutions of PCP are non-empty words.

**Remark 1.** For every instance  $(\Sigma, \sigma, \tau)$  of PCP,  $(\Sigma, \sigma, \tau)$  is a yes-instance of PCP if and only if there exists  $a \in \Sigma$  such that  $(\Sigma, \sigma, \tau, \sigma(a), \varepsilon, \tau(a), \varepsilon)$  is a yes-instances GPCP.

For each integer  $k \geq 1$ , GPCP( $k$ ) denotes the restriction of GPCP to instances  $(\Sigma, \sigma, \tau, s, t, s', t')$  such that  $\Sigma$  has cardinality  $k$ .

## 1.2 Semi-Thue systems

Formally, a *semi-Thue system* is a pair  $T = (\Sigma, R)$ , where  $\Sigma$  is an alphabet and where  $R$  is a subset of  $\Sigma^* \times \Sigma^*$ . The elements of  $R$  are called the *rules* of  $T$ . For every  $x, y \in \Sigma^*$ , we say that  $y$  is *immediately derivable* from  $x$  modulo  $T$ , and we write  $x \mapsto_T y$ , if there exist  $s, t, z, z' \in \Sigma^*$  such that  $x = zsz'$ ,  $y = ztz'$  and  $(s, t) \in R$ . For every  $u, v \in \Sigma^*$ , we say that  $u$  is *derivable* from  $v$  modulo  $T$ , and we write  $u \xrightarrow{*}_T v$ , if there exist an integer  $n \geq 0$  and  $x_0, x_1, \dots, x_n \in \Sigma^*$  such that  $x_0 = u$ ,  $x_n = v$ , and  $x_{i-1} \mapsto_T x_i$  for every  $i \in [1, n]$ :

$$u = x_0 \xrightarrow{T} x_1 \xrightarrow{T} x_2 \xrightarrow{T} \dots \xrightarrow{T} x_n = v. \quad (1)$$

In other words,  $\xrightarrow{*}_T$  is the reflexive-transitive closure of the binary relation  $\mapsto_T$ . Define the ACCESSIBILITY problem as: given a semi-Thue system  $T$  and two words  $u$  and  $v$  over

the alphabet of  $T$ , decide whether  $u \xrightarrow{\star}_T v$ . For every integer  $k \geq 1$ , define  $\text{ACCESSIBILITY}(k)$  as the restriction of  $\text{ACCESSIBILITY}$  to instances  $(T, u, v)$  such that  $T$  has  $k$  rules.

### 1.3 Decidability

Let  $k$  be a positive integer. The decidabilities of  $\text{ACCESSIBILITY}$ ,  $\text{PCP}$  and  $\text{GPCP}$  are linked by the following four facts.

**Fact 1.** *If  $\text{GPCP}(k)$  is decidable then  $\text{PCP}(k)$  is decidable.*

**Fact 2.** *If  $\text{GPCP}(k+2)$  is decidable then  $\text{ACCESSIBILITY}(k)$  is decidable.*

**Fact 3.** *If  $\text{PCP}(k+2)$  is decidable then  $\text{GPCP}(k)$  is decidable.*

**Fact 4** (Claus's theorem). *If  $\text{PCP}(k+4)$  is decidable then  $\text{ACCESSIBILITY}(k)$  is decidable.*

Fact 1 follows from Remark 1, Fact 3 is [7, Theorem 3.2], and Fact 4 was originally stated by Claus [2, Theorem 2] (see also [6, 8, 7]). To our knowledge, Fact 2 is explicitly stated for the first time in the present paper.

**Remark 2.** *The conjunction of Facts 2 and 3 yields Fact 4.*

Since Matiyasevich and Sénizergues have shown that  $\text{ACCESSIBILITY}(3)$  is undecidable [9, Theorem 4.1], it follows from Fact 4 that  $\text{PCP}(7)$  is undecidable [9, Corollary 1]. In the same way Fact 2 yields that  $\text{GPCP}(5)$  is undecidable (see also [6, Theorem 7]). Those results are the sharpest to date. Indeed, the decidability of each of the following eight problems is still open:

- $\text{ACCESSIBILITY}(1)$ ,  $\text{ACCESSIBILITY}(2)$ ,
- $\text{GPCP}(3)$ ,  $\text{GPCP}(4)$ ,
- $\text{PCP}(3)$ ,  $\text{PCP}(4)$ ,  $\text{PCP}(5)$  and  $\text{PCP}(6)$ .

However, Ehrenfeucht and Rozenberg showed that  $\text{PCP}(2)$  and  $\text{GPCP}(2)$  are decidable [4] (see also [3, 5]).

### 1.4 Organization of the paper

The aim of the paper is to present a clean, detailed proof of Fact 4. We start in Section 2 with some technicalities concerning  $\text{ACCESSIBILITY}$ . Then, Fact 2 is proved in Section 3, and Fact 3 is proved in Section 4.

## 2 More on the decidability of Accessibility

**Definition 1.** The language  $\{010^n101 : n \geq 2\}$  is denoted by  $C$ . For each integer  $k \geq 1$ , define  $\mathcal{C}_k$  as the set of all instances  $(T, u, v)$  of ACCESSIBILITY such that  $u, v \in C^*$  and  $T = (\{0, 1\}, R)$  for some  $k$ -element subset  $R \subseteq C^+ \times C^+$ .

The essential properties of the gadget language  $C$  are:  $C$  is an infinite, binary, comma-free code (see Definitions 5 and 6 below), and no word in  $C$  overlaps the delimiter word 0011.

The aim of this section is to show:

**Proposition 1.** For every integer  $k \geq 1$ , the general ACCESSIBILITY( $k$ ) problem is decidable if and only if its restriction to  $\mathcal{C}_k$  is decidable.

The idea to prove Proposition 1 is to construct a many-one reduction based on the following gadget transformation:

**Definition 2.** Let  $T = (\Sigma, R)$  be a semi-Thue system, let  $\Delta$  be an alphabet, and let  $\alpha : \Sigma^* \rightarrow \Delta^*$ . Define the image of  $T$  under  $\alpha$ , denoted  $\alpha(T)$ , as the semi-Thue system over  $\Delta$  with rule set  $\{(\alpha(s), \alpha(t)) : (s, t) \in R\}$ .

The next two lemmas are straightforward.

**Lemma 1.** Let  $\Sigma$  and  $\widehat{\Sigma}$  be alphabets, let  $T$  be a semi-Thue system over  $\Sigma$ , let  $\widehat{T}$  be a semi-Thue system over  $\widehat{\Sigma}$ , and let  $\alpha : \Sigma^* \rightarrow \widehat{\Sigma}^*$  be such that for every  $x, y \in \Sigma^*$ ,  $x \mapsto_T y$  implies  $\alpha(x) \mapsto_{\widehat{T}} \alpha(y)$ . For every  $u, v \in \Sigma^*$ ,  $u \xrightarrow{*}_T v$  implies  $\alpha(u) \xrightarrow{*}_{\widehat{T}} \alpha(v)$ .

*Proof.* Assume that  $u \xrightarrow{*}_T v$ : there exist an integer  $n \geq 0$  and  $n + 1$  words  $x_0, x_1, \dots, x_n$  over  $\Sigma$  such that Equation (1) holds. It follows

$$\alpha(u) = \alpha(x_0) \xrightarrow{\widehat{T}} \alpha(x_1) \xrightarrow{\widehat{T}} \alpha(x_2) \xrightarrow{\widehat{T}} \dots \xrightarrow{\widehat{T}} \alpha(x_n) = \alpha(v), \quad (2)$$

and thus  $\alpha(u) \xrightarrow{*}_{\widehat{T}} \alpha(v)$ . □

**Lemma 2.** In the notation of Definition 2, if  $\alpha$  is a morphism then for every  $u, v \in \Sigma^*$ ,  $u \xrightarrow{*}_T v$  implies  $\alpha(u) \xrightarrow{*}_{\alpha(T)} \alpha(v)$ .

*Proof.* We apply Lemma 1 with  $\widehat{T} := \alpha(T)$ . Let  $x, y \in \Sigma^*$  be such that  $x \mapsto_T y$ : there exist  $s, t, z, z' \in \Sigma^*$  such that  $x = zsz'$ ,  $y = ztz'$  and  $(s, t) \in R$ . Since  $\alpha$  is a morphism,  $\alpha(x)$  and  $\alpha(y)$  can be parsed as follows:  $\alpha(x) = \alpha(z)\alpha(s)\alpha(z')$ ,  $\alpha(y) = \alpha(z)\alpha(t)\alpha(z')$  and  $(\alpha(s), \alpha(t))$  is a rule of  $\alpha(T)$ . Hence, we get that  $\alpha(x) \mapsto_{\alpha(T)} \alpha(y)$ . □

**Definition 3.** Let  $(s, t)$  be a rule of some semi-Thue system:  $(s, t)$  is a pair of words. We say that  $(s, t)$  is an insertion rule if  $s = \varepsilon$ . We say that  $(s, t)$  is a deletion rule if  $t = \varepsilon$ . A semi-Thue system is called  $\varepsilon$ -free if it has neither insertion nor deletion rule. By extension, an instance  $(T, u, v)$  of ACCESSIBILITY is called  $\varepsilon$ -free if the semi-Thue system  $T$  is  $\varepsilon$ -free.

Note that every instance of  $\text{ACCESSIBILITY}(k)$  that belongs to  $\mathcal{C}_k$  is  $\varepsilon$ -free. The next two gadget morphisms play crucial roles in the proofs of both Lemma 3 and Theorem 2 below.

**Definition 4.** Let  $\Sigma$  be an alphabet and let  $d$  be a letter. Define  $\lambda_d$  and  $\rho_d$  as the morphisms from  $\Sigma^*$  to  $(\Sigma \cup \{d\})^*$  given by:  $\lambda_d(a) := da$  and  $\rho_d(a) := ad$  for every  $a \in \Sigma$ .

For instance,  $\lambda_d(01101)$  and  $\rho_d(01101)$  equal  $d0d1d1d0d1$  and  $0d1d1d0d1d$ , respectively.

**Lemma 3.** For every integer  $k \geq 1$ ,  $\text{ACCESSIBILITY}(k)$  is decidable if and only if the problem is decidable on  $\varepsilon$ -free instances.

*Proof.* We present a many-one reduction from  $\text{ACCESSIBILITY}(k)$  in its general form to  $\text{ACCESSIBILITY}(k)$  on  $\varepsilon$ -free instances.

Let  $(T, u, v)$  be an instance of  $\text{ACCESSIBILITY}(k)$ . Let  $\Sigma$  denote the alphabet  $T$ , let  $d$  be a symbol such that  $d \notin \Sigma$ , and let  $\mu : \Sigma^* \rightarrow (\Sigma \cup \{d\})^*$  be defined by:  $\mu(w) := \lambda_d(w)d = d\rho_d(w)$  for every  $w \in \Sigma^*$ . Clearly,  $(\mu(T), \mu(u), \mu(v))$  is an  $\varepsilon$ -free instance of  $\text{ACCESSIBILITY}(k)$ , and  $(\mu(T), \mu(u), \mu(v))$  is computable from  $(T, u, v)$ .

It remains to check the correctness statement:  $u \xrightarrow{*}_T v$  if and only if  $\mu(u) \xrightarrow{*}_{\mu(T)} \mu(v)$ .

(only if). Let  $x, y \in \Sigma^*$  be such that  $x \xrightarrow{*}_T y$ : there exist  $s, t, z, z' \in \Sigma^*$  such that  $x = zsz'$ ,  $y = ztz'$  and  $(s, t)$  is a rule of  $T$ . Clearly,  $\mu(x)$  and  $\mu(y)$  can be parsed as follows:  $\mu(x) = \lambda_d(z)\mu(s)\rho_d(z')$ ,  $\mu(y) = \lambda_d(z)\mu(t)\rho_d(z')$  and  $(\mu(s), \mu(t))$  is a rule of  $\mu(T)$ . Hence, we get that  $\mu(x) \xrightarrow{*}_{\mu(T)} \mu(y)$ . It now follows from Lemma 1 (applied with  $\alpha := \mu$  and  $\hat{T} := \mu(T)$ ) that  $u \xrightarrow{*}_T v$  implies  $\mu(u) \xrightarrow{*}_{\mu(T)} \mu(v)$ .

(if). let  $\hat{\mu} : (\Sigma \cup \{d\})^* \rightarrow \Sigma^*$  denote the morphism defined by:  $\hat{\mu}(a) := a$  for every  $a \in \Sigma$  and  $\hat{\mu}(d) := \varepsilon$ . It is clear that  $\hat{\mu}(\mu(w)) = w$  for every  $w \in \Sigma^*$ , and thus  $T = \hat{\mu}(\mu(T))$ . Hence, for every  $\hat{u}, \hat{v} \in (\Sigma \cup \{d\})^*$ ,  $\hat{u} \xrightarrow{*}_{\mu(T)} \hat{v}$  implies  $\hat{\mu}(\hat{u}) \xrightarrow{*}_T \hat{\mu}(\hat{v})$  by Lemma 2 (applied with  $\alpha := \hat{\mu}$  and  $T := \mu(T)$ ). In particular,  $\mu(u) \xrightarrow{*}_{\mu(T)} \mu(v)$  implies  $u = \hat{\mu}(\mu(u)) \xrightarrow{*}_T \hat{\mu}(\mu(v)) = v$ .

□

Given an alphabet  $\Sigma$ , a semi-Thue system  $T$  over  $\Sigma$ , and a subset  $L \subseteq \Sigma^*$ , we say that  $L$  is *closed under derivation modulo  $T$*  if for every  $x \in L$  and every  $y \in \Sigma^*$ ,  $x \xrightarrow{*}_T y$  implies  $y \in L$ . The next lemma is an *ad hoc* counterpart of Lemma 1.

**Lemma 4.** Let  $\Sigma$  and  $\hat{\Sigma}$  be alphabets, let  $T$  be a semi-Thue system over  $\Sigma$ , let  $\hat{T}$  be a semi-Thue system over  $\hat{\Sigma}$ , and let  $\alpha : \Sigma^* \rightarrow \hat{\Sigma}^*$  be such that:

- (i) the range of  $\alpha$  is closed under derivation modulo  $\hat{T}$ , and
- (ii) for every  $x, y \in \Sigma^*$ ,  $\alpha(x) \xrightarrow{*}_{\hat{T}} \alpha(y)$  implies  $x \xrightarrow{*}_T y$ .

For every  $u, v \in \Sigma^*$ ,  $\alpha(u) \xrightarrow{*}_{\hat{T}} \alpha(v)$  implies  $u \xrightarrow{*}_T v$ .

*Proof.* Assume that  $\alpha(u) \xrightarrow{\star}_{\hat{T}} \alpha(v)$ : there exist an integer  $n \geq 0$  and  $n + 1$  words  $\hat{x}_0, \hat{x}_1, \dots, \hat{x}_n$  over  $\hat{\Sigma}$  such that

$$\alpha(u) = \hat{x}_0 \xrightarrow{\hat{T}} \hat{x}_1 \xrightarrow{\hat{T}} \hat{x}_2 \xrightarrow{\hat{T}} \dots \xrightarrow{\hat{T}} \hat{x}_n = \alpha(v).$$

It follows from point (i) that  $\hat{x}_i$  belongs to the range of  $\alpha$  for every  $i \in [0, n]$ : let  $x_0 := u$ , let  $x_n := v$ , and for each  $i \in [1, n - 1]$ , let  $x_i \in \Sigma^*$  be such that  $\hat{x}_i = \alpha(x_i)$ . Now, Equation (2) holds, and thus Equation (1) follows by point (ii). We have thus shown that  $u \xrightarrow{\star}_T v$ .  $\square$

Surprisingly, hypothesis (i) of Lemma 4 is not disposable. Indeed, let  $T = (\Sigma, R)$  be a semi-Thue system, and let  $u_0, v_0 \in \Sigma^*$  be such that  $u_0 \not\xrightarrow{\star}_T v_0$ : a trivial choice for  $R$  is the empty set. Let  $a$  be a symbol such that  $a \notin \Sigma$ , and let  $\hat{T} := (\Sigma \cup \{a\}, R \cup \{(u_0, a), (a, v_0)\})$ . For every  $x, y \in \Sigma^*$ ,  $x \xrightarrow{\star}_T y$  is equivalent to  $x \xrightarrow{\star}_{\hat{T}} y$ . However,  $\Sigma^*$  is not closed under derivation modulo  $\hat{T}$ , and  $u \xrightarrow{\star}_{\hat{T}} v$  does not imply  $u \xrightarrow{\star}_T v$  for every  $u, v \in \Sigma^*$ , since  $u_0 \xrightarrow{\star}_{\hat{T}} v_0$ .

**Definition 5.** Let  $X$  be a language. We say that  $X$  is a code if the property

$$x_1 x_2 \dots x_m = y_1 y_2 \dots y_n \iff (x_1, x_2, \dots, x_m) = (y_1, y_2, \dots, y_n)$$

holds for any integers  $m, n \geq 1$  and any elements  $x_1, x_2, \dots, x_m, y_1, y_2, \dots, y_n \in X$ .

Note that  $(x_1, x_2, \dots, x_m) = (y_1, y_2, \dots, y_n)$  means that both  $m = n$  and  $x_i = y_i$  for every  $i \in [1, m]$ . In other words, a language  $X$  is a code if each word in  $X^*$  has a unique factorization over  $X$ . A morphism  $\alpha : \Sigma^* \rightarrow \Delta^*$  is injective if and only if  $\alpha$  is injective on  $\Sigma$  and  $\alpha(\Sigma)$  is a code.

**Definition 6** ([1]). A code  $X$  is called comma-free if for every words  $x, z$  and  $z'$ ,

$$(x \in X \text{ and } xz z' \in X^*) \implies (z \in X^* \text{ and } z' \in X^*).$$

Every comma-free code is a *bifix* code: no word in the language is a prefix or a suffix of another word in the language. For instance,  $K := \{10^n 1 : n \geq 1\}$  and  $C$  are comma-free codes, but  $K \cup \{11\}$  is a bifix code which is not comma-free.

**Lemma 5.** In the notation of Definition 2, assume that

- (i)  $\alpha$  is an injective morphism,
- (ii)  $\alpha(\Sigma)$  is a comma-free code, and
- (iii)  $T$  has no insertion rule.

For every  $u, v \in \Sigma^*$ ,  $u \xrightarrow{\star}_T v$  is equivalent to  $\alpha(u) \xrightarrow{\star}_{\alpha(T)} \alpha(v)$ .

*Proof.* According to Lemma 2,  $u \xrightarrow{*}_T v$  implies  $\alpha(u) \xrightarrow{*}_{\alpha(T)} \alpha(v)$ . Conversely, let us prove that  $\alpha(u) \xrightarrow{*}_{\alpha(T)} \alpha(v)$  implies  $u \xrightarrow{*}_T v$ . We rely on Lemma 4.

Let  $\hat{x}, \hat{y} \in \Delta^*$  be such that  $\hat{x}$  belongs to the range of  $\alpha$  and  $\hat{x} \xrightarrow{*}_{\alpha(T)} \hat{y}$ : there exist  $\hat{z}, \hat{z}' \in \Delta^*$  and  $(s, t) \in R$  such that  $\alpha(\hat{x}) = \hat{z}\alpha(s)\hat{z}'$  and  $\hat{y} = \hat{z}\alpha(t)\hat{z}'$ . Since  $\alpha$  is a morphism, the range of  $\alpha$  equals  $\alpha(\Sigma)^*$ . In particular, both  $\alpha(s)$  and  $\hat{z}\alpha(s)\hat{z}'$  belong to  $\alpha(\Sigma)^*$ . Furthermore,  $\alpha(s)$  belongs to  $\alpha(\Sigma)^+$ : indeed,  $s$  is a non-empty word because  $T$  has no insertion rule, and thus its image under the injective morphism  $\alpha$  is also a non-empty word. It follows that both  $\hat{z}$  and  $\hat{z}'$  belong to  $\alpha(\Sigma)^*$  because  $\alpha(\Sigma)$  is a comma-free code: there exist  $z, z' \in \Sigma^*$  such that  $\alpha(z) = \hat{z}$  and  $\alpha(z') = \hat{z}'$ . We can now write  $\hat{x}$  and  $\hat{y}$  in the forms  $\hat{x} = \alpha(zsz')$  and  $\hat{y} = \alpha(ztz')$ . Hence,  $\hat{y}$  belongs to the range of  $\alpha$ , which proves that the range of  $\alpha$  is closed under derivation modulo  $\alpha(T)$ . Moreover, we also get  $\alpha^{-1}(\hat{x}) = zsz' \xrightarrow{*}_T ztz' = \alpha^{-1}(\hat{y})$ . Therefore, Lemma 4 applies with  $\hat{T} := \alpha(T)$ .  $\square$

Let us thoroughly examine the hypotheses of Lemma 5. Hypothesis (iii) could be replaced with “ $T$  has no deletion rule”: apply Lemma 5 in its original form to the reversal of  $T$ , defined as  $\tilde{T} := (\Sigma, \{(t, s) : (s, t) \in R\})$ . However, the following two counterexamples show that neither hypothesis (ii) nor hypothesis (iii) is disposable.

**Counterexample 1.** Let  $T := (\{a, b\}, \{(a, aa)\})$  and let  $\alpha : \{a, b\}^* \rightarrow \{0, 1\}^*$  be the morphism defined by  $\alpha(a) := 01$  and  $\alpha(b) := 011$ :  $\alpha(T) = (\{0, 1\}, \{(01, 0101)\})$ . Clearly,  $\alpha$  is injective and  $T$  is  $\varepsilon$ -free. However,  $\alpha(\{a, b\})$  is not a comma-free code, and  $\alpha(u) \xrightarrow{*}_{\alpha(T)} \alpha(v)$  does not imply  $u \xrightarrow{*}_T v$  for every  $u, v \in \{a, b\}^*$ :  $\alpha(b) \xrightarrow{*}_{\alpha(T)} \alpha(ab)$  but  $b \not\xrightarrow{*}_T ab$ .

Counterexample 1 disproves a claim from Claus’s original paper [2, page 57, line –4]. A statement from Harju, Karhumäki and Krob [8, page 43, line 1] is disproved in the same way.

**Counterexample 2.** Let  $T := (\{a, b, c\}, \{(\varepsilon, a), (b, \varepsilon)\})$  and let  $\alpha : \{a, b, c\}^* \rightarrow \{0, 1\}^*$  be the morphism defined by  $\alpha(a) := 101$ ,  $\alpha(b) := 1001$  and  $\alpha(c) := 10001$ :  $\alpha(T) = (\{0, 1\}, \{(\varepsilon, 101), (1001, \varepsilon)\})$ . Clearly,  $\alpha$  is injective and  $\alpha(\{a, b, c\})$  is a comma-free code. However,  $T$  admits both insertion and deletion rules, and  $\alpha(u) \xrightarrow{*}_{\alpha(T)} \alpha(v)$  does not imply  $u \xrightarrow{*}_T v$  for every  $u, v \in \{a, b\}^*$ :  $c \not\xrightarrow{*}_T a$  but

$$\alpha(c) \xrightarrow{\alpha(T)} 10101001 \xrightarrow{\alpha(T)} 10101001011 \xrightarrow{\alpha(T)} 1010011 \xrightarrow{\alpha(T)} \alpha(a).$$

*Proof of Proposition 1.* We present a many-one reduction from ACCESSIBILITY( $k$ ) on  $\varepsilon$ -free instances to ACCESSIBILITY( $k$ ) on  $\mathcal{C}_k$ , so that Lemma 3 applies.

Let  $(T, u, v)$  be an  $\varepsilon$ -free instance of ACCESSIBILITY( $k$ ). Let  $\Sigma$  denote the alphabet of  $T$ . Compute an injection  $\alpha : \Sigma \rightarrow C$ . The morphism from  $\Sigma^*$  to  $\{0, 1\}$  that extends  $\alpha$  is also denoted  $\alpha$ . Clearly,  $(\alpha(T), \alpha(u), \alpha(v))$  belongs to  $\mathcal{C}_k$ , and  $(\alpha(T), \alpha(u), \alpha(v))$  is computable from  $(T, u, v)$ . Moreover,  $u \xrightarrow{*}_T v$  is equivalent to  $\alpha(u) \xrightarrow{*}_{\alpha(T)} \alpha(v)$  by Lemma 5.  $\square$

### 3 From GPCP to Accessibility

The key ingredient of the proof of Fact 2 is the *accordion lemma* (Lemma 7 below).

**Definition 7.** A word  $f$  is called *bordered* if there exist three non-empty words  $u$ ,  $v$  and  $x$  such that  $f = xu = vx$ .

Equivalently,  $f$  is bordered if and only if there exists a word  $z$  with  $|z| < 2|f|$  such that  $f$  occurs twice or more in  $z$ .

**Definition 8.** We say that two words  $x$  and  $y$  *overlap* if at least one of the following four assertions hold:

- (i)  $x$  occurs in  $y$ ,
- (ii)  $y$  occurs in  $x$ ,
- (iii) some non-empty prefix of  $x$  is a suffix of  $y$ , or
- (iv) some non-empty prefix of  $y$  is a suffix of  $x$ .

Since we make the convention that the empty word occurs in every word, the empty word and any other word do overlap. Two non-empty words  $x$  and  $y$  overlap if and only if there exists a word  $z$  with  $|z| < |x| + |y|$  such that both  $x$  and  $y$  occur in  $z$ .

We can now state a protoversion of the accordion lemma.

**Lemma 6.** Let  $T = (\Sigma, R)$  be a semi-Thue system, and let  $f, u, v \in \Sigma^*$  be such that:

- (i)  $f$  is unbordered,
- (ii)  $f$  does not occur in  $u$ ,
- (iii)  $f$  does not occur in  $v$ , and
- (iv) for each rule  $(s, t) \in R$ ,  $s$  and  $f$  do not overlap.

Then,  $u \xrightarrow{*}_T v$  holds if and only if there exist  $x, y \in \Sigma^*$  satisfying both  $xfv = ufy$  and  $x \xrightarrow{*}_T y$ .

*Proof.* (only if). If  $u \xrightarrow{*}_T v$  then  $x := u$  and  $y := v$  are such that  $xfv = ufy$  and  $x \xrightarrow{*}_T y$ .

(if). Assume that there exist  $x, y \in \Sigma^*$  such that  $xfv = ufy$  and  $x \xrightarrow{*}_T y$ . Let  $n$  denote the number of occurrences of  $f$  in  $xfv$ . Since  $f$  is unbordered (hypothesis (i)), those occurrences are pairwise non-overlapping:

$$xfv = ufy = w_0fw_1fw_2 \cdots fw_n$$



for some words  $w_0, w_1, \dots, w_n \in \Sigma^*$ . Since  $f$  does not occur in  $v$  (hypothesis (iii)), we have  $v = w_n$  and

$$x = w_0 f w_1 f w_2 \cdots f w_{n-1}.$$

In the same way,  $f$  does not occur in  $u$  either (hypothesis (ii)), and thus we have also  $u = w_0$  and

$$y = w_1 f w_2 f w_3 \cdots f w_n.$$

Now, remark that  $f$  plays the role of a delimiter with respect to the derivation modulo  $T$  (hypothesis (iv)):  $x \xrightarrow{\star}_T y$  implies

$$w_{i-1} \xrightarrow{\star}_T w_i$$

for every  $i \in [1, n]$ . Therefore,  $u \xrightarrow{\star}_T v$  holds. □

Let us comment the statement of Lemma 6. Hypothesis (iv) implies that  $T$  has no insertion rule. It could be replaced with “for every  $(s, t) \in R$ ,  $t$  and  $f$  do not overlap”. Hypothesis (ii) is in fact disposable: the verification is left to the reader. Let  $\Sigma$  be an alphabet, let  $f$  be a symbol such that  $f \notin \Sigma$ , and let  $T$  be a semi-Thue system over  $\Sigma \cup \{f\}$  with rules in  $\Sigma^+ \times \Sigma^*$ . An easy consequence of Lemma 6 is that, for every  $u, v \in \Sigma^*$ ,  $u \xrightarrow{\star}_T v$  holds if and only if there exist  $x, y \in (\Sigma \cup \{f\})^*$  satisfying both  $xfv = ufy$  and  $x \xrightarrow{\star}_T y$ .

**Definition 9.** *The word 0011 is denoted by  $f$ .*

**Lemma 7** (Accordion lemma). *Let  $k$  be a positive integer. For every  $(T, u, v) \in \mathcal{C}_k$ ,  $(T, u, v)$  is a yes-instance of ACCESSIBILITY( $k$ ) if and only if there exist  $x, y \in \{0, 1\}^*$  satisfying both  $xfv = ufy$  and  $x \xrightarrow{\star}_T y$ .*

*Proof.* Clearly,  $f$  is unbordered, and for every  $s \in C^+$ ,  $s$  and  $f$  do not overlap. Hence, Lemma 6 applies. □

The statement of the accordion lemma can be made precise as follows (the verification is left to the reader): for any  $(T, u, v) \in \mathcal{C}_k$  and any  $x, y \in \{0, 1\}^*$  such that  $xfv = ufy$  and  $x \xrightarrow{\star}_T y$ , both  $x$  and  $y$  belong to  $(C \cup \{f\})^*$ . Besides, if  $C$  and  $f$  were defined as  $C := \{10^n 1 : n \geq 1\}$  and  $f := 11$  in Definitions 1 and 9, then Proposition 1 and Lemma 7 would hold (the verification is left to the reader). In [7, Theorem 4.1], Harju and Karhumäki present a proof of Claus’s theorem that implicitly relies on those variants of Proposition 1 and Lemma 7.

**Definition 10.** *An instance  $(\Sigma, \sigma, \tau, s, t, s', t')$  of GPCP is called erasement-free if  $\sigma(\Sigma) \cup \tau(\Sigma) \subseteq \{0, 1\}^+$ .*

We can now prove a slightly strengthened version of Fact 2.

**Theorem 1.** *Let  $k$  be a positive integer. If  $\text{GPCP}(k+2)$  is decidable on erasement-free instances then  $\text{ACCESSIBILITY}(k)$  is decidable.*

*Proof.* In order to apply Proposition 1, we present a many-one reduction from  $\text{ACCESSIBILITY}(k)$  on  $\mathcal{C}_k$  to  $\text{GPCP}(k+2)$  on erasement-free instances.

Let  $(T, u, v)$  be an element of  $\mathcal{C}_k$ : there exist  $s_1, s_2, \dots, s_k, t_1, t_2, \dots, t_k \in C^+$  such that  $T = (\{0, 1\}, \{(s_1, t_1), (s_2, t_2), \dots, (s_k, t_k)\})$ .

Let  $a_1, a_2, \dots, a_k$  be  $k$  symbols such that  $\Sigma := \{0, 1, a_1, a_2, \dots, a_k\}$  is an alphabet of cardinality  $k+2$ . Let  $\sigma, \tau : \Sigma^* \rightarrow \{0, 1\}^*$  be the morphisms defined by:

$$\begin{aligned} \sigma(0) &:= 0, & \tau(0) &:= 0, \\ \sigma(1) &:= 1, & \tau(1) &:= 1, \\ \sigma(a_i) &:= s_i, & \tau(a_i) &:= t_i \end{aligned}$$

for every  $i \in [1, k]$ . Let  $J$  denote the instance  $(\Sigma, \sigma, \tau, \varepsilon, fv, uf, \varepsilon)$  of  $\text{GPCP}(k+2)$ .

It is clear that  $J$  is erasement-free and that  $J$  is computable from  $I$ . It remains to check that  $I$  is a yes-instance of  $\text{ACCESSIBILITY}(k)$  if and only if  $J$  is a yes-instance of  $\text{GPCP}(k+2)$ . The proof of the “if part” relies on the accordion lemma while the proof of the “only if part” relies on next lemma.

**Lemma 8.** *For any  $x, y \in \{0, 1\}^*$  such that  $x \xrightarrow{T} y$ , there exists  $z \in \Sigma^*$  such that  $x = \sigma(z)$  and  $y = \tau(z)$ .*

*Proof.* Let  $z', z'' \in \{0, 1\}^*$  and let  $i \in [1, k]$  be such that  $x = z's_iz''$  and  $y = z't_iz''$ . A suitable choice for  $z$  is  $z'a_iz''$ .  $\square$

(only if). Assume that  $u \xrightarrow{T}^* v$ . There exist an integer  $n \geq 0$  and  $n+1$  words  $x_0, x_1, \dots, x_n$  over  $\{0, 1\}$  such that Equation (1) holds. Lemma 8 ensures that there exists  $z_i \in \Sigma^*$  satisfying  $x_{i-1} = \sigma(z_i)$  and  $x_i = \tau(z_i)$  for each  $i \in [1, n]$ . Now,  $w := z_1 f z_2 f z_3 \cdots f z_n$  is such that  $\sigma(w)fv = uf\tau(w)$ .

(if). Assume that there exists  $w \in \Sigma^*$  such that  $\sigma(w)fv = uf\tau(w)$ . The morphisms  $\sigma$  and  $\tau$  are defined in such a way that  $\sigma(z) \xrightarrow{T}^* \tau(z)$  for every  $z \in \Sigma^*$ . In particular,  $x := \sigma(w)$  and  $y := \tau(w)$  are such that  $xfv = ufy$  and  $x \xrightarrow{T}^* y$ . Hence, Lemma 7 yields  $u \xrightarrow{T}^* v$ .  $\square$

Combining Theorem 1 and [9, Theorem 4.1] we obtain:

**Corollary 1.**  *$\text{GPCP}(5)$  is undecidable on erasement-free instances.*

## 4 From PCP to GPCP

**Definition 11.** An instance  $(\Sigma, \sigma, \tau, s, t, s', t')$  of GPCP is called  $(\varepsilon, \varepsilon)$ -free if for every  $a \in \Sigma$ ,  $(\sigma(a), \tau(a)) \neq (\varepsilon, \varepsilon)$ .

**Lemma 9.** For every integer  $k \geq 1$ , GPCP( $k$ ) is decidable if and only if the problem is decidable on  $(\varepsilon, \varepsilon)$ -free instances.

*Proof.* We present a many-one reduction from GPCP( $k$ ) to GPCP( $k$ ) on  $(\varepsilon, \varepsilon)$ -free instances.

Let  $I := (\Sigma, \sigma, \tau, s, t, s', t')$  be an instance of GPCP( $k$ ). Compute the set  $\widehat{\Sigma}$  of all letters  $a \in \Sigma$  such that  $(\sigma(a), \tau(a)) \neq (\varepsilon, \varepsilon)$ . If  $\widehat{\Sigma}$  is empty then solving GPCP( $k$ ) on  $I$  reduces to checking whether  $st$  and  $s't'$  are equal. Hence, we may assume  $\widehat{\Sigma} \neq \emptyset$  without loss of generality, taking out of the way cumbersome considerations. Let  $\widehat{\sigma}$  and  $\widehat{\tau}$  denote the restrictions to  $\widehat{\Sigma}^*$  of  $\sigma$  and  $\tau$ , respectively. Let  $J$  denote the septuple  $(\widehat{\Sigma}, \widehat{\sigma}, \widehat{\tau}, s, t, s', t')$ . Clearly,  $J$  is an  $(\varepsilon, \varepsilon)$ -free instance of GPCP( $k$ ) and  $J$  is computable from  $I$ . Moreover,  $I$  is a yes-instance of GPCP( $k$ ) if and only if  $J$  is also a yes-instance of the problem.  $\square$

Remark that every erasement-free instance of GPCP is  $(\varepsilon, \varepsilon)$ -free, but the converse is false in general.

**Definition 12.** An instance  $(\Sigma, \sigma, \tau)$  of PCP is called erasement-free if  $\sigma(\Sigma) \cup \tau(\Sigma) \subseteq \{0, 1\}^+$ .

We can now prove Fact 3.

**Theorem 2.** Let  $k$  be a positive integer.

- (i). If PCP( $k + 2$ ) is decidable then GPCP( $k$ ) is decidable.
- (ii). If PCP( $k + 2$ ) is decidable on erasement-free instances then GPCP( $k$ ) is decidable on erasement-free instances.

*Proof.* We present a many-one reduction from GPCP( $k$ ) on  $(\varepsilon, \varepsilon)$ -free instances to PCP( $k + 2$ ) in order to apply Lemma 9.

Let  $I := (\Sigma, \sigma, \tau, s, t, s', t')$  be an  $(\varepsilon, \varepsilon)$ -free instance of GPCP( $k$ ). Without loss of generality, we may assume  $\mathbf{b} \notin \Sigma$  and  $\mathbf{e} \notin \Sigma$ :  $\widehat{\Sigma} := \Sigma \cup \{\mathbf{b}, \mathbf{e}\}$  is an alphabet of cardinality  $k + 2$ . Let  $\lambda := \lambda_{\mathbf{a}}$  and  $\rho := \rho_{\mathbf{a}}$  (see Definition 4). Let  $\widehat{\sigma}, \widehat{\tau} : \widehat{\Sigma}^* \rightarrow \{0, 1, \mathbf{d}, \mathbf{b}, \mathbf{e}\}^*$  be the two morphisms defined by:

$$\begin{aligned} \widehat{\sigma}(\mathbf{b}) &:= \mathbf{b}\lambda(s), & \widehat{\tau}(\mathbf{b}) &:= \mathbf{b}\mathbf{d}\rho(s'), \\ \widehat{\sigma}(\mathbf{e}) &:= \lambda(t)\mathbf{d}\mathbf{e}, & \widehat{\tau}(\mathbf{e}) &:= \rho(t')\mathbf{e}, \\ \widehat{\sigma}(a) &:= \lambda(\sigma(a)), & \widehat{\tau}(a) &:= \rho(\tau(a)) \end{aligned}$$

for every  $a \in \Sigma$ . Let  $j : \{0, 1, \mathbf{d}, \mathbf{b}, \mathbf{e}\}^* \rightarrow \{0, 1\}^*$  denote an injective morphism: for instance  $j$  can be given by  $j(0) := 000$ ,  $j(1) := 111$ ,  $j(\mathbf{d}) := 101$ ,  $j(\mathbf{b}) := 100$  and  $j(\mathbf{e}) := 001$ .

It is clear that  $J := (\widehat{\Sigma}, j \circ \widehat{\sigma}, j \circ \widehat{\tau})$  is an instance of  $\text{PCP}(k+2)$  computable from  $I$ , and that  $J$  is erasement-free whenever  $I$  is erasement-free. Hence, to prove both points (i) and (ii) of Theorem 2, it remains to check that  $I$  is a yes-instance of  $\text{GPCP}(k)$  if and only if  $J$  is a yes-instance of  $\text{PCP}(k+2)$ .

**Lemma 10.** *For every  $w \in \Sigma^*$ ,  $s\sigma(w)t = s'\tau(w)t'$  if and only if  $\widehat{\sigma}(\mathbf{b}w\mathbf{e}) = \widehat{\tau}(\mathbf{b}w\mathbf{e})$ .*

*Proof.* Straightforward computations yield

$$\widehat{\sigma}(\mathbf{b}w\mathbf{e}) = \widehat{\sigma}(\mathbf{b})\widehat{\sigma}(w)\widehat{\sigma}(\mathbf{e}) = \mathbf{b}\lambda(s)\lambda(\sigma(w))\lambda(t)\mathbf{d}\mathbf{e} = \mathbf{b}\lambda(s\sigma(w)t)\mathbf{d}\mathbf{e}$$

and

$$\widehat{\tau}(\mathbf{b}w\mathbf{e}) = \widehat{\tau}(\mathbf{b})\widehat{\tau}(w)\widehat{\tau}(\mathbf{e}) = \mathbf{b}\mathbf{d}\rho(s')\rho(\tau(w))\rho(t')\mathbf{d}\mathbf{e} = \mathbf{b}\mathbf{d}\rho(s'\tau(w)t')\mathbf{e}.$$

Since  $\lambda(x)\mathbf{d} = \mathbf{d}\rho(x)$  for every  $x \in \{0,1\}^*$ ,  $s\sigma(w)t = s'\tau(w)t'$  implies  $\widehat{\sigma}(\mathbf{b}w\mathbf{e}) = \widehat{\tau}(\mathbf{b}w\mathbf{e})$ , and furthermore,  $\widehat{\sigma}(\mathbf{b}w\mathbf{e}) = \widehat{\tau}(\mathbf{b}w\mathbf{e})$  implies  $\lambda(s\sigma(w)t) = \lambda(s'\tau(w)t')$ . Since  $\lambda$  is trivially injective,  $\widehat{\sigma}(\mathbf{b}w\mathbf{e}) = \widehat{\tau}(\mathbf{b}w\mathbf{e})$  implies  $s\sigma(w)t = s'\tau(w)t'$ .  $\square$

If  $I$  is a yes instance of  $\text{GPCP}(k)$  then it follows from Lemma 10 that  $J$  is a yes-instance of  $\text{PCP}(k+2)$ . The converse is slightly more complicated to prove.

**Lemma 11.** *For every  $w \in \widehat{\Sigma}^*$ , the following three assertions are equivalent:*

1.  $\widehat{\sigma}(w\mathbf{e})$  is a prefix of  $\widehat{\tau}(w\mathbf{e})$ ,
2.  $\widehat{\tau}(w\mathbf{e})$  is a prefix of  $\widehat{\sigma}(w\mathbf{e})$ , and
3.  $\widehat{\sigma}(w\mathbf{e}) = \widehat{\tau}(w\mathbf{e})$ .

*Proof.* The letter  $\mathbf{e}$  occurs once in  $\widehat{\sigma}(\mathbf{e})$  (resp.  $\widehat{\tau}(\mathbf{e})$ ) whereas for every  $a \in \Sigma \cup \{\mathbf{b}\}$ ,  $\mathbf{e}$  does not occur at all in  $\widehat{\sigma}(a)$  (resp.  $\widehat{\tau}(a)$ ). Therefore,  $|\widehat{\sigma}(x)|_{\mathbf{e}} = |x|_{\mathbf{e}} = |\widehat{\tau}(x)|_{\mathbf{e}}$  holds for every  $x \in \widehat{\Sigma}^*$ . Since  $\mathbf{e}$  is the last letter of  $\widehat{\sigma}(\mathbf{e})$ , any proper prefix of  $\widehat{\sigma}(w\mathbf{e})$  contains less occurrences of  $\mathbf{e}$  than  $\widehat{\tau}(w\mathbf{e})$ . From that we deduce that  $\widehat{\tau}(w\mathbf{e})$  cannot be a proper prefix of  $\widehat{\sigma}(w\mathbf{e})$ . In the same way,  $\widehat{\sigma}(w\mathbf{e})$  cannot be a proper prefix of  $\widehat{\tau}(w\mathbf{e})$ .  $\square$

**Lemma 12.** *For every  $w \in \widehat{\Sigma}^*$ , the following three assertions are equivalent:*

1.  $\widehat{\sigma}(\mathbf{b}w)$  is a suffix of  $\widehat{\tau}(\mathbf{b}w)$ ,
2.  $\widehat{\tau}(\mathbf{b}w)$  is a suffix of  $\widehat{\sigma}(\mathbf{b}w)$ , and
3.  $\widehat{\sigma}(\mathbf{b}w) = \widehat{\tau}(\mathbf{b}w)$ .

*Proof.* Lemma 12 is proved in the same way as Lemma 11. The details are left to the reader.  $\square$

**Claim 1.** *Let  $a \in \widehat{\Sigma}$  be such that  $\widehat{\sigma}(a) \neq \varepsilon$ .*

- (i). *The first letter of  $\widehat{\sigma}(a)$  is either  $\mathbf{b}$  or  $\mathbf{d}$ .*

(ii). The last letter of  $\hat{\sigma}(a)$  is distinct from  $\mathbf{d}$ .

**Claim 2.** Let  $a \in \hat{\Sigma}$  be such that  $\hat{\tau}(a) \neq \varepsilon$ .

(i). The first letter of  $\hat{\tau}(a)$  is distinct from  $\mathbf{d}$ .

(ii). The last letter of  $\hat{\tau}(a)$  is either  $\mathbf{d}$  or  $\mathbf{e}$ .

Assume that  $J$  is a yes-instance of  $\text{PCP}(k+2)$ . Let  $w \in \hat{\Sigma}^+$  be such that  $\hat{\sigma}(w) = \hat{\tau}(w)$ . Let  $x$  denote both words  $\hat{\sigma}(w)$  and  $\hat{\tau}(w)$ .

Since  $I$  is an  $(\varepsilon, \varepsilon)$ -free instance of GPCP,  $(\hat{\sigma}(a), \hat{\tau}(a))$  is distinct from  $(\varepsilon, \varepsilon)$  for every  $a \in \hat{\Sigma}$ , and thus  $x$  is a non-empty word. Combining Claims 1(i) and 2(i), we obtain that  $\mathbf{b}$  is the first letter of  $x$ , and thus  $\mathbf{b}$  is also the first letter of  $w$ . In the same way, combining Claims 1(ii) and 2(ii), we obtain that  $\mathbf{e}$  is the last letter of  $x$ , and thus  $\mathbf{e}$  is also the last letter of  $w$ . Hence,  $w$  is of the form  $\mathbf{b}w'\mathbf{e}$  with  $w' \in \hat{\Sigma}^*$ .

Now, assume that  $w$  is a *shortest* non-empty word over  $\hat{\Sigma}$  such that  $\hat{\sigma}(w) = \hat{\tau}(w)$ . Let us check that  $w' \in \Sigma^*$ . By the way of contradiction suppose that  $\mathbf{e}$  occurs in  $w'$ : there exist  $w_1, w_2 \in \hat{\Sigma}^*$  such that  $w' = w_1\mathbf{e}w_2$ . Straightforward computations yield  $\hat{\sigma}(\mathbf{b}w_1\mathbf{e})\hat{\sigma}(w_2\mathbf{e}) = x = \hat{\tau}(\mathbf{b}w_1\mathbf{e})\hat{\tau}(w_2\mathbf{e})$ . Therefore,  $\hat{\sigma}(\mathbf{b}w_1\mathbf{e})$  is a prefix of  $\hat{\tau}(\mathbf{b}w_1\mathbf{e})$  or  $\hat{\tau}(\mathbf{b}w_1\mathbf{e})$  is a prefix of  $\hat{\sigma}(\mathbf{b}w_1\mathbf{e})$ . From Lemma 11, we deduce that  $\hat{\sigma}(\mathbf{b}w_1\mathbf{e}) = \hat{\tau}(\mathbf{b}w_1\mathbf{e})$ . Since  $\mathbf{b}w_1\mathbf{e}$  is shorter than  $w$ , a contradiction follows. Hence  $\mathbf{e}$  does not occur in  $w'$ . Similar arguments based on Lemma 12 show that  $\mathbf{b}$  does not occur in  $w'$  either.

Hence,  $w'$  is a word over  $\Sigma$ , and thus Lemma 10 ensures that  $s\sigma(w')t = s'\tau(w')t'$ . It follows that  $I$  is a yes-instance of  $\text{GPCP}(k)$ .  $\square$

Strictly speaking, the correspondence problem that was originally introduced by Post in his 1946 paper [10] is, in our terminology, the restriction of PCP to erasement-free instances.

Combining Theorems 1 and 2(ii), we obtain a slightly strengthened version of Claus's theorem (Fact 4).

**Corollary 2.** Let  $k$  be a positive integer. If  $\text{PCP}(k+4)$  is decidable on erasement-free instances then  $\text{ACCESSIBILITY}(k)$  is decidable.

Combining Corollary 2 and [9, Theorem 4.1] we obtain:

**Corollary 3.**  $\text{PCP}(7)$  is undecidable on erasement-free instances.

## References

- [1] J. Berstel and D. Perrin. *Theory of Codes*. Pure and Applied Mathematics. Academic Press, 1985.
- [2] V. Claus. Some remarks on  $\text{PCP}(k)$  and related problems. *The Bulletin of the European Association for Theoretical Computer Science (EATCS)*, 12:54 – 61, 1980.

- [3] A. Ehrenfeucht, J. Karhumäki, and G. Rozenberg. The (generalized) Post correspondence problem with lists consisting of two words is decidable. *Theoretical Computer Science*, 21(2):119–144, 1982.
- [4] A. Ehrenfeucht and G. Rozenberg. On the (generalized) Post correspondence problem with lists of length 2. In S. Even and O. Kariv, editors, *Proceedings of the 8th International Colloquium on Automata, Languages and Programming (ICALP'81)*, volume 115 of *Lecture Notes in Computer Science*, pages 408–416. Springer, 1981.
- [5] V. Halava, T. Harju, and M. Hirvensalo. Binary (generalized) Post correspondence problem. *Theoretical Computer Science*, 276(1–2):183–204, 2002.
- [6] V. Halava, T. Harju, and M. Hirvensalo. Undecidability bounds for integer matrices using Claus instances. *International Journal of Foundations of Computer Science*, 18(5):931–948, 2007.
- [7] T. Harju and J. Karhumäki. Morphisms. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 1, pages 439–510. Springer, 1997.
- [8] T. Harju, J. Karhumäki, and D. Krob. Remarks on generalized Post correspondence problem. In C. Puech and R. Reischuk, editors, *Proceedings of the 13th Annual Symposium on Theoretical Aspects of Computer Science (STACS'96)*, volume 1046 of *Lecture Notes in Computer Science*, pages 39–48. Springer, 1996.
- [9] Y. Matiyasevich and G. Sénizergues. Decision problems for semi-Thue systems with a few rules. *Theoretical Computer Science*, 330(1):145–169, 2005.
- [10] E. L. Post. A variant of a recursively unsolvable problem. *Bulletin of the American Mathematical Society*, 52(4):264–268, 1946.